Automating a Campus with Cisco NSO

Amy Liebowitz, University of Michigan

UNIVERSITY OF
MICHIGAN

Outline

- Overview of Automation at UM
- Why NSO?
- Quick NSO Overview
- Campus Service Design
- Configuration Example
- On-Boarding Buildings into NSO
- NetDash Integration
- Lessons Learned
- Questions

University of Michigan Automation - Overview

- **Goal: Automate the entire campus network with a single configuration source of truth (i.e. "source of intent").**
- Major campus network refresh project provides unique opportunity.
  - Project encompasses entire network: edge, core, distribution, data center.
  - New devices will be installed in parallel (as opposed to rip-and-replace).
  - First major greenfield deployment in over 10 years.
- Automation strategically coupled with refresh.
  - All new devices are fully automated.
  - Migrated buildings have automated access layer as well.
- Currently around 35% of routers and switches are automated (1700 out of 4600).

Why Cisco NSO?

- New network will be primarily composed of Cisco NXOS and IOS devices.
- IOS and NXOS have significant limitations when attempting “byo automation”.
    - CLI is designed to be interactive (as opposed to stateless/RESTful).
    - Limited or no native candidate config/rollback features.
    - yang/netconf implementation not well-supported.
- Cisco NSO:
    - Is a product fully supported by Cisco.
    - Supports many non-Cisco platforms (at least for now).
    - Scalable and extensible enough to automate the entire campus.

NSO Overview - Device Manager

- NSO stores all network device configurations in one database.
- Database is a tree-like structure defined with YANG.
- Network Element Drivers (NEDs) convert device configurations into YANG-defined structured data.
  - NEDs exist for many different network vendors.
  - Enables staging, comparing, and rolling back configuration changes on devices that don’t support this natively (namely IOS and NXOS).
- Changes on multiple devices can be implemented with a single commit to the database.
  - If a single failure is detected, changes on all devices are rolled back.
  - Can change this behavior with different commit options.

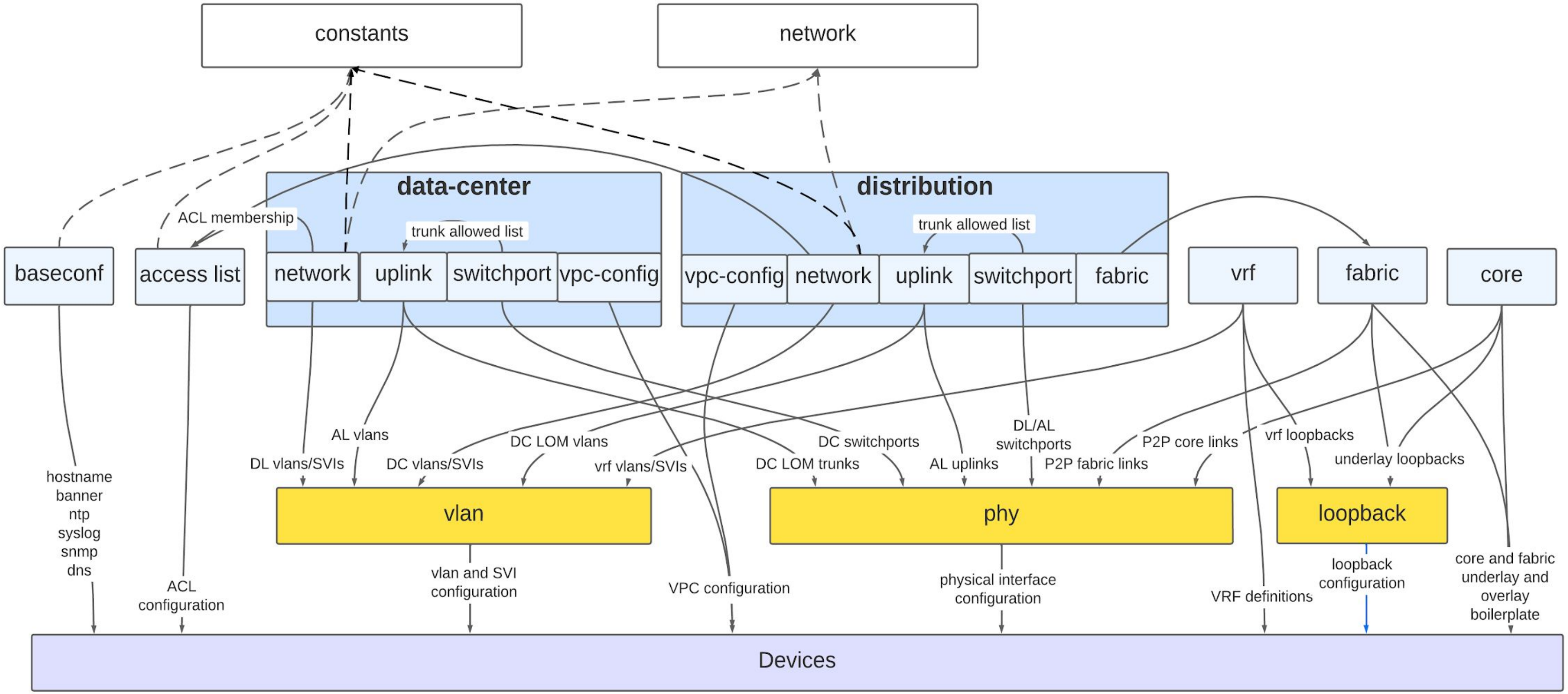
NSO Overview - Service Manager

- Services are custom abstractions of network features - things like “VRF”, “switchport”, or “access list”.
- You define your own services in YANG based on what makes sense for your organization.
- You write code that maps service data to device configuration.
    - Code applies custom XML templates that reference NED settings to drive device configuration.
    - NSO provides code and template skeletons to work from.
- When templates are applied, NSO calculates the difference between desired and existing device configuration.
    - NSO pushes the minimum number of commands needed to achieve desired state to the devices.
    - “Reverse diff” is saved so NSO can back out changes when service is deleted.

Campus Service Design

- **Three major service types**
  - High Level Services
    - User-facing abstractions of major network components.
    - Most complex high level service is “distribution”.
      - Models all aspects of a building network.
  - Low Level Services
    - Hides platform-specific complexities from higher level services.
    - Only configured by higher level services (“service stacking”) - hidden from the CLI.
    - Most complex low level service is “phy”.
      - Models all aspects of physical port configuration.
  - Data Only Services
    - Stores structured data used by other services.
    - Changing this data does not trigger any configuration changes on the network.

Campus Service Design



→ Service A writes data to Service B (stacking)

⇢ Service A reads data from Service B

□ Data Only Service

🟦 High Level Service

🟨 Low Level Service

Example - Configuring a Building Network

General Steps

1. Define details in data-only network service.
   - Subnet(s), VLAN ID, VRF, DHCP Relay Servers, ACLs.
2. Tie network to a building in the distribution service.
   - Service code configures the network on the building distribution routers.
3. Configure access ports.
   - Service code defines VLAN on switches, adds it to trunk allowed lists, and configures access ports.

Example - Configuring a Building Network

Step 1: Define network in data-only network service

```
admin@ncs% show | compare
 services {
+    network V-TEST-NETWORK {
+        role user;
+        layer3 {
+            vrf PRIMARY;
+            primary-ipv4-subnet 10.255.0.0/24;
+            dhcp-relay-servers CORE1-DHCP-SERVERS;
+            ingress-acl ANTISPOOF-IN;
+            egress-acl ANTISPOOF-OUT;
+        }
+        layer2 {
+            vlan-id 50;
+        }
+    }
 }
[ok][2023-09-07 15:02:18]
```

Example - Configuring a Building Network

Step 2: Add Network to Building (Distribution Zone)

```
admin@ncs% set services distribution bldga network V-TEST-NETWORK
[ok][2023-09-07 15:03:19]
admin@ncs% commit dry-run outformat native
native {
    device {
        name dl-bldga-1
        data ip access-list VLAN50-IP-IN
              10 permit udp any any eq bootps
              ... [ output omitted ] ...

             vlan 50
              name V-TEST-NETWORK
             exit
             interface Vlan50
              no shutdown
              description V-TEST-NETWORK
              vrf member PRIMARY
              ip access-group VLAN50-IP-IN in
              ip access-group VLAN50-IP-OUT out
              ip address 10.255.0.2/24
              ip dhcp relay address 141.211.147.229
             ... [ output omitted ] ...
```

Example - Configuring a Building Network

Step 3: Configure access ports

```
admin@ncs% set services distribution bldga switch al-bldga-1 switchport Gi1/3 description
"Test user" mode access vlan V-TEST-NETWORK
...
admin@ncs% commit dry-run outformat native
native {
    device {
        name al-bldga-1
        data vlan 50
              name V-TEST-NETWORK
             !
             interface Port-channel1
              switchport trunk allowed vlan 50
             exit
             interface GigabitEthernet1/3
              no shutdown
              switchport
              switchport mode access
              switchport access vlan 50
              description "Test user"
               ... [ output omitted ] ...
```

On-Boarding into NSO

- Approximately one building a week is migrated to the new core.
- Migration has three phases:
  - On-boarding
    - New distribution routers are brought online and connected to the new core.
  - Pre-migration
    - Network service data is populated from the existing router configuration.
  - Migration
    - Temporary trunk built between old and new routers.
    - SVIs and loopbacks migrated from old routers to new.
    - Switchport and uplink service data is generated.
    - Switch uplinks are physically re-cabled.
    - Old routers are removed from service.

On-Boarding into NSO

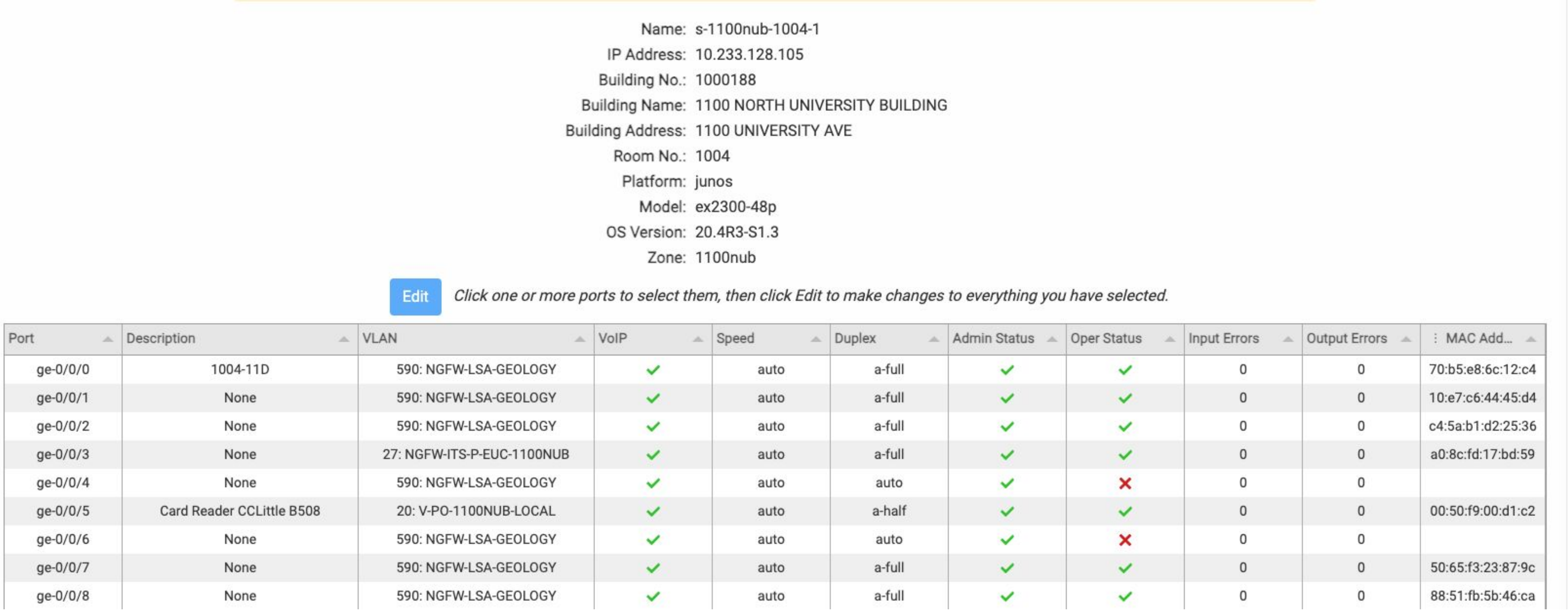
- NSO Actions are heavily leveraged during the migration.
  - Actions are meant to effect a one-way change (no “reverse diff” is saved).
  - Like services, structure of an action is defined with yang and implemented in code.
  - Actions are invoked from the CLI (or via netconf/restconf)
  - NSO has many built-in actions (eg “sync-from”, “fetch-ssh-host-keys”).
- General migration automation strategy:
  - Use actions to on-board building devices into NSO.
  - Use more actions to translate NSO device configuration data into service data.
    - Device configuration data is already structured - config parsing has never been easier.
    - Actions also pull data from external sources (google sheets, IPAM, etc).

UNIVERSITY OF MICHIGAN

NetDash Integration

- We support a custom web application that enables unit IT to make access port changes in buildings.
  - Legacy app is called “Device Configuration Tool” (DCT).
    - Reads and writes directly to switches.
    - Changes made in DCT cause sync issues with devices managed by NSO.
    - Written in perl, original developer is retired.
- New tool called NetDash has been developed to replace DCT.
  - Django app, much easier for developers to support.
    - Reads and writes to NSO via NETCONF.
    - Developed dedicated NSO Actions for this application.
    - Currently being augmented to support data center switches.
- Buildings on-boarded into NSO are disabled in DCT - users are directed to NetDash

NetDash Integration



Name: s-1100nub-1004-1
IP Address: 10.233.128.105
Building No.: 1000188
Building Name: 1100 NORTH UNIVERSITY BUILDING
Building Address: 1100 UNIVERSITY AVE
Room No.: 1004
Platform: junos
Model: ex2300-48p
OS Version: 20.4R3-S1.3
Zone: 1100nub

Edit *Click one or more ports to select them, then click Edit to make changes to everything you have selected.*

Port	Description	VLAN	VoIP	Speed	Duplex	Admin Status	Oper Status	Input Errors	Output Errors	MAC Add...
ge-0/0/0	1004-11D	590: NGFW-LSA-GEOLOGY	✓	auto	a-full	✓	✓	0	0	70:b5:e8:6c:12:c4
ge-0/0/1	None	590: NGFW-LSA-GEOLOGY	✓	auto	a-full	✓	✓	0	0	10:e7:c6:44:45:d4
ge-0/0/2	None	590: NGFW-LSA-GEOLOGY	✓	auto	a-full	✓	✓	0	0	c4:5a:b1:d2:25:36
ge-0/0/3	None	27: NGFW-ITS-P-EUC-1100NUB	✓	auto	a-full	✓	✓	0	0	a0:8c:fd:17:bd:59
ge-0/0/4	None	590: NGFW-LSA-GEOLOGY	✓	auto	auto	✓	✗	0	0	
ge-0/0/5	Card Reader CCLittle B508	20: V-PO-1100NUB-LOCAL	✓	auto	a-half	✓	✓	0	0	00:50:f9:00:d1:c2
ge-0/0/6	None	590: NGFW-LSA-GEOLOGY	✓	auto	auto	✓	✗	0	0	
ge-0/0/7	None	590: NGFW-LSA-GEOLOGY	✓	auto	a-full	✓	✓	0	0	50:65:f3:23:87:9c
ge-0/0/8	None	590: NGFW-LSA-GEOLOGY	✓	auto	a-full	✓	✓	0	0	88:51:fb:5b:46:ca

Lessons Learned

- Don’t try to design for every use case
  - Predicting the future is hard
  - You don’t have to support everything initially
  - Augmenting services later is easier than attempting to unravel complexity in production
  - Low-touch, one-off configurations can be left out of service design
    - As long as the NED supports the configuration you can manage these changes with NSO device manager.
    - But only if a service won’t overwrite the configuration.

Lessons Learned

- Servicepoint placement is important in NSO
  - Servicepoints trigger code execution when any data at or below in the tree is changed.
  - Servicepoint evaluates all data, not just what has changed.
  - Break servicepoint into multiple smaller ones that live further down the tree to increase performance and decrease individual servicepoint complexity.
  - Originally we had a servicepoint that addressed any change on an access layer switch.
    - Since changed to several servicepoints that handle changes at a port level.
    - Need an action that re-deploys all ports on a switch as a result.

Q & A

amylieb@umich.edu

UNIVERSITY OF
MICHIGAN